PAPI – (Performance Application Programming Interface)

Kevin London

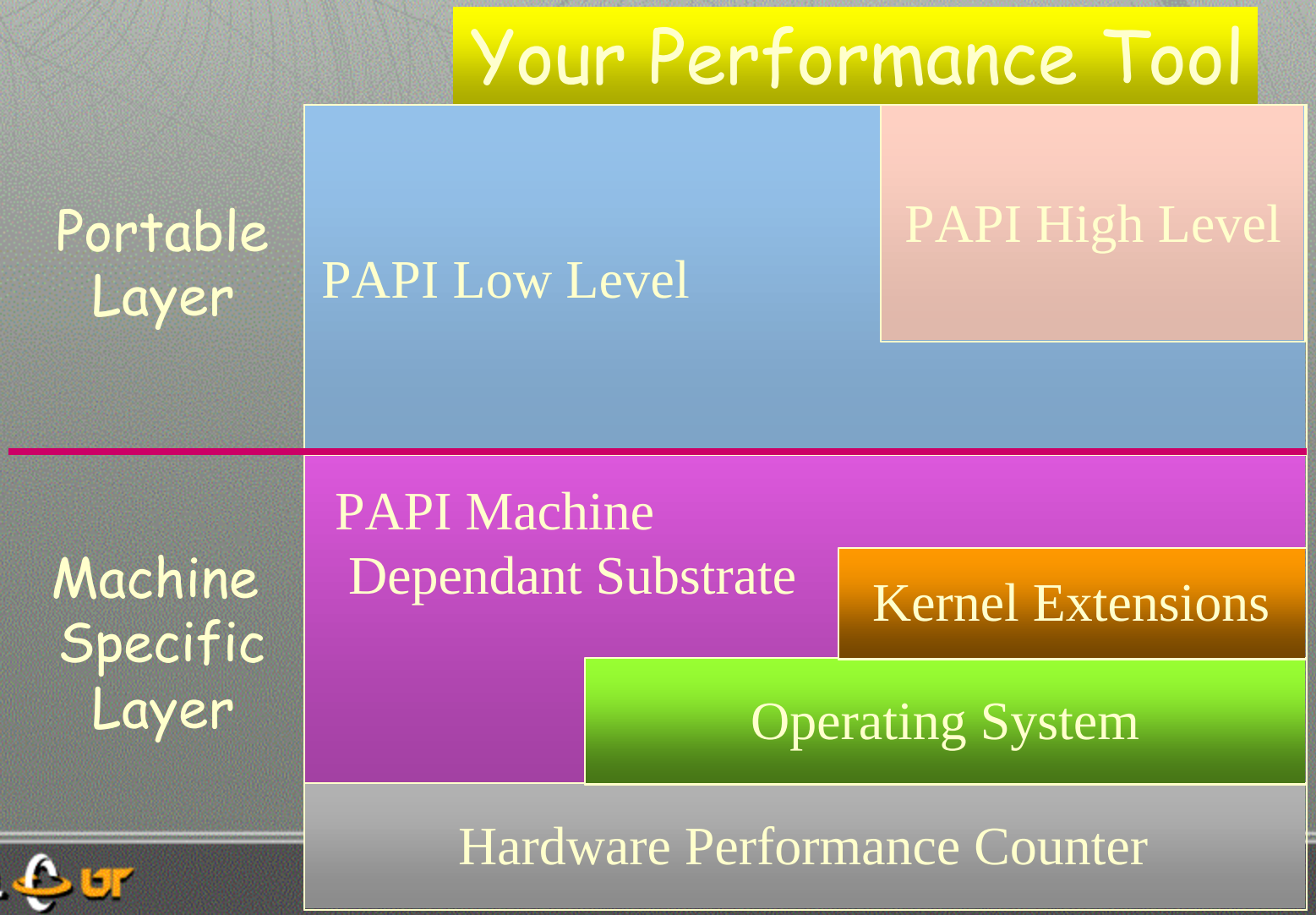
May 27, 2003

Innovative Computing Laboratory
COMPUTER SCIENCE DEPARTMENT
UNIVERSITY OF TENNESSEE

What is PAPI?

- PAPI is a portable interface that provides access machine specific counters that monitor performance information.
- Implemented on Power 3 & 4, Cray T3E, AMD Athlon, Intel Pentium Pro-Pentium 4, Itanium 1 & 2, UltraSparc I,II,III, Mips R10000,R12000 and R14000 and Alpha processors. With Cray X1 and AMD Opteron support underway.

PAPI **Implementation**



PAPI Counter Interfaces

- PAPI provides three interfaces to the underlying counter hardware:
  1. The low level interface manages hardware events in user defined groups called *EventSets*.
  2. The high level interface simply provides the ability to start, stop and read the counters for a specified list of events.
  3. Graphical tools to visualize information.

High-level Interface

- Meant for application programmers wanting coarse-grained measurements
- Not thread safe
- Calls the lower level API
- Allows only PAPI preset events
- Easier to use and less setup (additional code) than low-level

High-level API

- C interface
  PAPI_start_counters
  PAPI_read_counters
  PAPI_stop_counters
  PAPI_accum_counters
  PAPI_num_counters
  PAPI_flops
- Fortran interface
  PAPIF_start_counters
  PAPIF_read_counters
  PAPIF_stop_counters
  PAPIF_accum_counters
  PAPIF_num_counters
  PAPIF_flops

Low-Level Interface

- There's about 40 functions
- Increased efficiency and functionality over the high level PAPI interface
- Obtain information about the executable and the hardware.
- Thread safe
- Callbacks
- Multiplexing support allows more events than counters to be monitored even if the hardware does not support it.
- Overflow monitoring with function callbacks
- PAPI defined presets as well as native event support
- Profiling support using hardware support when available.

PAPI at NCSA

- Information on where PAPI is installed at NCSA and how to link with PAPI as well as examples is available at: http://www.ncsa.uiuc.edu/UserInfo/Resources/Software/Tools/PAPI/
- PAPI is located in the following directory: /usr/apps/tools/papi
- Add the directory /usr/apps/tools/papi/man to your **MANPATH** environment variable if you want to have the "man" command find the PAPI manual pages

Setting up PAPI

- Download the latest distributions from: http://icl.cs.utk.edu/papi/
- Install 3rd party software/kernel patches if necessary.
- Make the distribution with the command: make –f Makefile.platform
- Make sure the distribution works by running the test-suite: ./run_tests.sh
- You are ready to start working with PAPI, detailed install instructions are in the INSTALL.txt file.

Setting up the High-level Interface

- `Int PAPI_num_counters(void)`
  - Initializes PAPI (if needed)
  - Returns number of hardware counters
- `int PAPI_start_counters(int *events, int len)`
  - Initializes PAPI (if needed)
  - Sets up an event set with the given counters
  - Starts counting in the event set
- `int PAPI_library_init(int version)`
  - Low-level routine implicitly called by above

Controlling the Counters

- `PAPI_stop_counters(long_long *vals, int  alen)`
  - Stop counters and put counter values in array
- `PAPI_accum_counters(long_long *vals, int  alen)`
  - Accumulate counters into array and reset
- `PAPI_read_counters(long_long *vals, int  alen)`
  - Copy counter values into array and reset counters
- `PAPI_flops(float *rtime, float *ptime,`
  `long_long *flpins, float *mflops)`
  - Wallclock time, process time, FP ins since start,
  - Mflop/s since last call

PAPI_flops

- int PAPI_flops(float *real_time, float *proc_time, long_long *flpins, float *mflops)
  - Only two calls needed, PAPI_flops before and after the code you want to monitor
  - real_time is the wall-clocktime between the two calls
  - proc_time is the "virtual" time or time the process was actually executing between the two calls (not as fine grained as real_time but better for longer measurements)
  - flpins is the total floating point instructions executed between the two calls
  - mflops is the Mflop/s rating between the two calls

PAPI High-level Example

```
long long values[NUM_EVENTS];
unsigned int Events[NUM_EVENTS]={PAPI_TOT_INS,PAPI_TOT_CYC};
 /* Start the counters */
 PAPI_start_counters((int*)Events,NUM_EVENTS);
 /* What we are monitoring? */
 do_work();
 /* Stop the counters and store the results in values */
 retval = PAPI_stop_counters(values,NUM_EVENTS);
```

For More Information

- Visit the website: http://icl.cs.utk.edu/projects/papi/
- Or join one of the two mailing lists setup for PAPI, subscribing instructions can be found at PAPI's website.
  - ptools-perfapi@ptools.org is a general discussion list for the PAPI software
  - perfapi-devel@ptools.org is a mailing list for developers of PAPI, performance tools and kernel patches. Interested hackers are welcomed. All the CVS log messages go here